

# (12) UK Patent Application (19) GB (11) 2 304 945 (13) A

(43) Date of A Publication 26.03.1997

(21) Application No 9616653.3

(22) Date of Filing 08.08.1996

(30) Priority Data

(31) 08519332 (32) 25.08.1995 (33) US

(71) Applicant(s)

International Business Machines Corporation

(Incorporated in USA - New York)

Armonk, New York 10504, United States of America

(72) Inventor(s)

Ronald Henry Jr. Jones

(74) Agent and/or Address for Service

R J Burt

IBM United Kingdom Limited, Intellectual  
Property Department, Mail Point 110, Hursley Park,  
WINCHESTER, Hampshire, SO21 2JN,  
United Kingdom

(51) INT CL<sup>6</sup>

G06F 3/033 9/44

(52) UK CL (Edition O )

G4A AKS APX

(56) Documents Cited

EP 0567699 A1 WO 95/08148 A1 WO 93/15455 A1  
US 5388264 A

(58) Field of Search

UK CL (Edition O ) G4A AKS APX

INT CL<sup>6</sup> G06F 3/023 3/033 9/44 17/30

On-line databases: WPI, COMPUTER, INSPEC

(54) An object-oriented interface controlling multimedia devices

(57) The present invention provides a method and apparatus which control a plurality of multimedia devices in response to a user's manipulations of icons 228-238 on an object-oriented graphical user interface 226. The user can connect and control the multimedia devices as desired by simply manipulating icons representative of the multimedia devices displayed on the graphical user interface. In particular the user can easily connect multimedia devices by graphically connecting their icons. The computer will then automatically perform the necessary connections between the corresponding objects, and each device can then be controlled through the user interface.

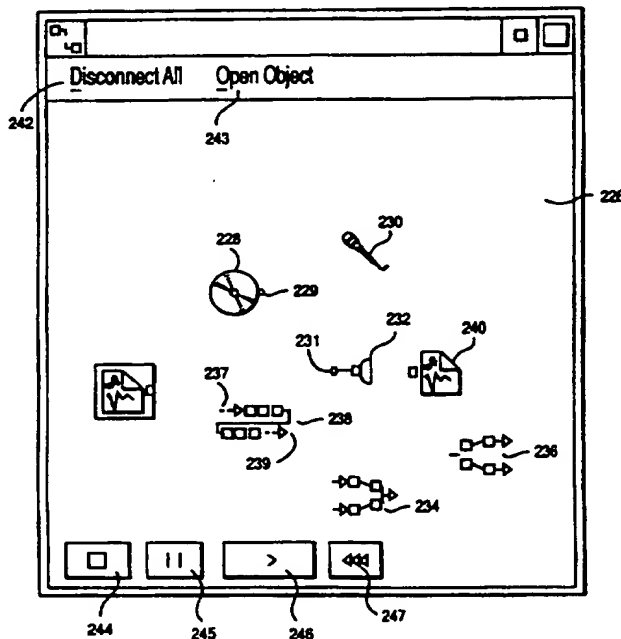


FIG. 2

GB 2 304 945 A

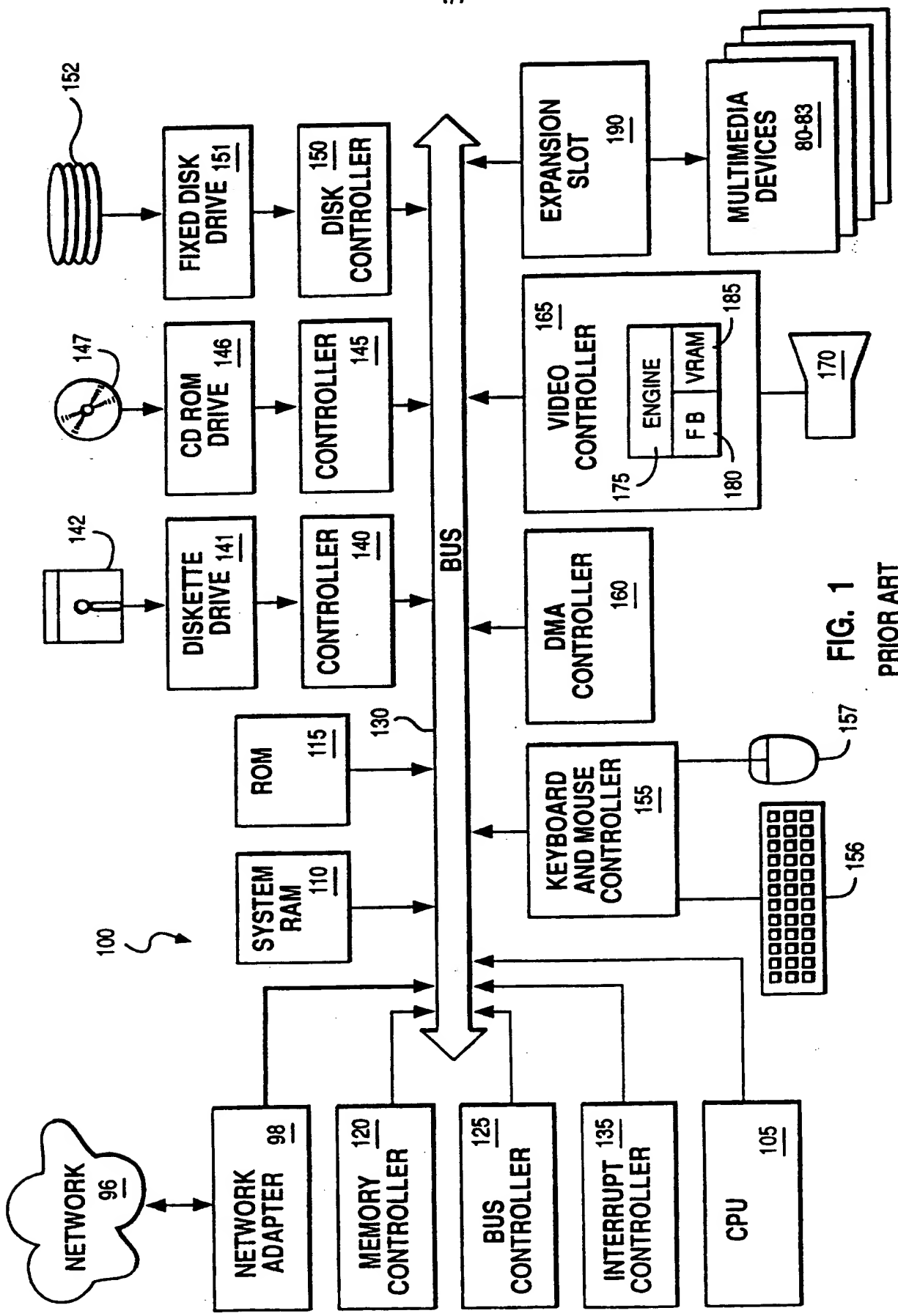


FIG. 1

PRIOR ART

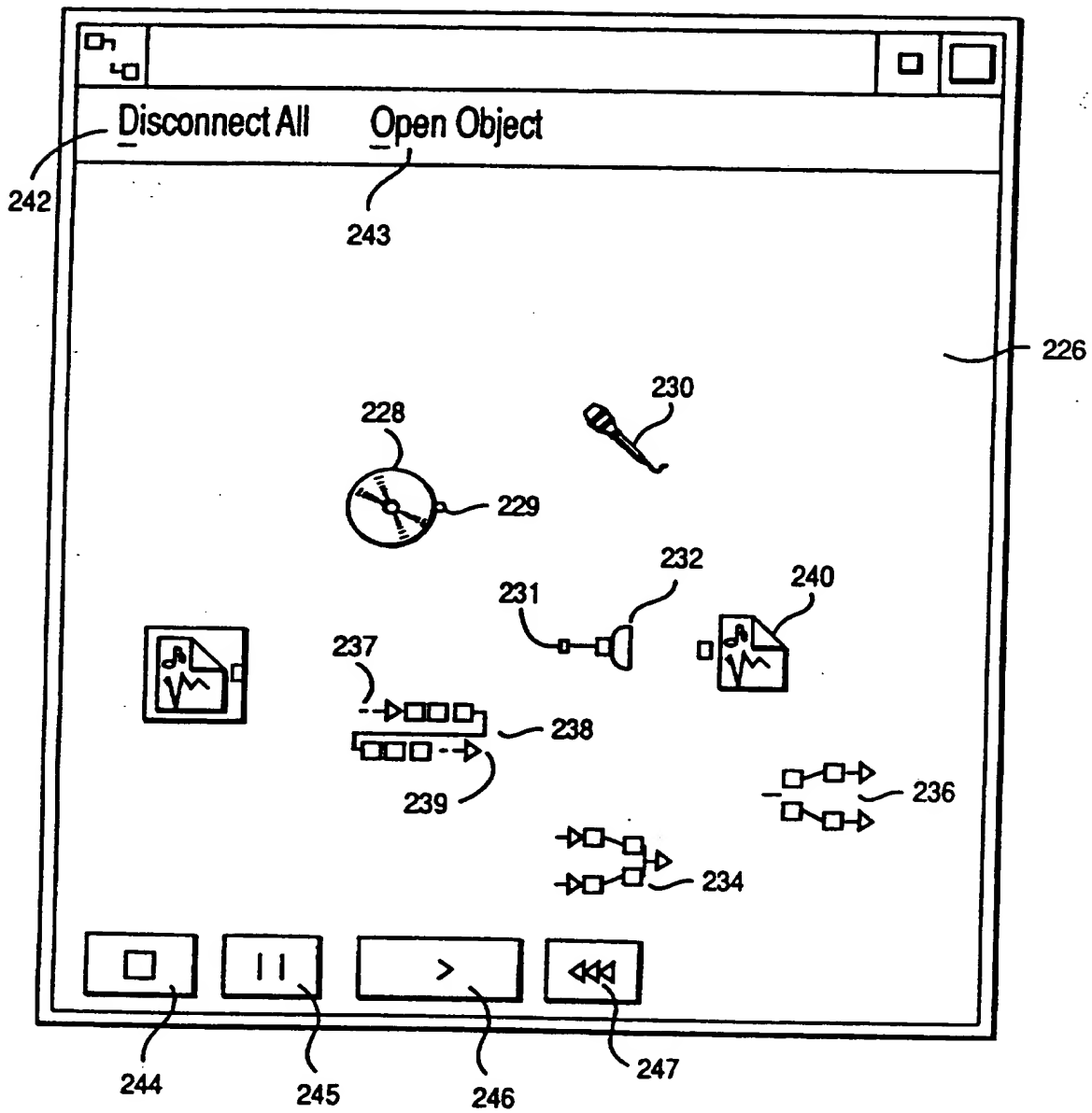


FIG. 2

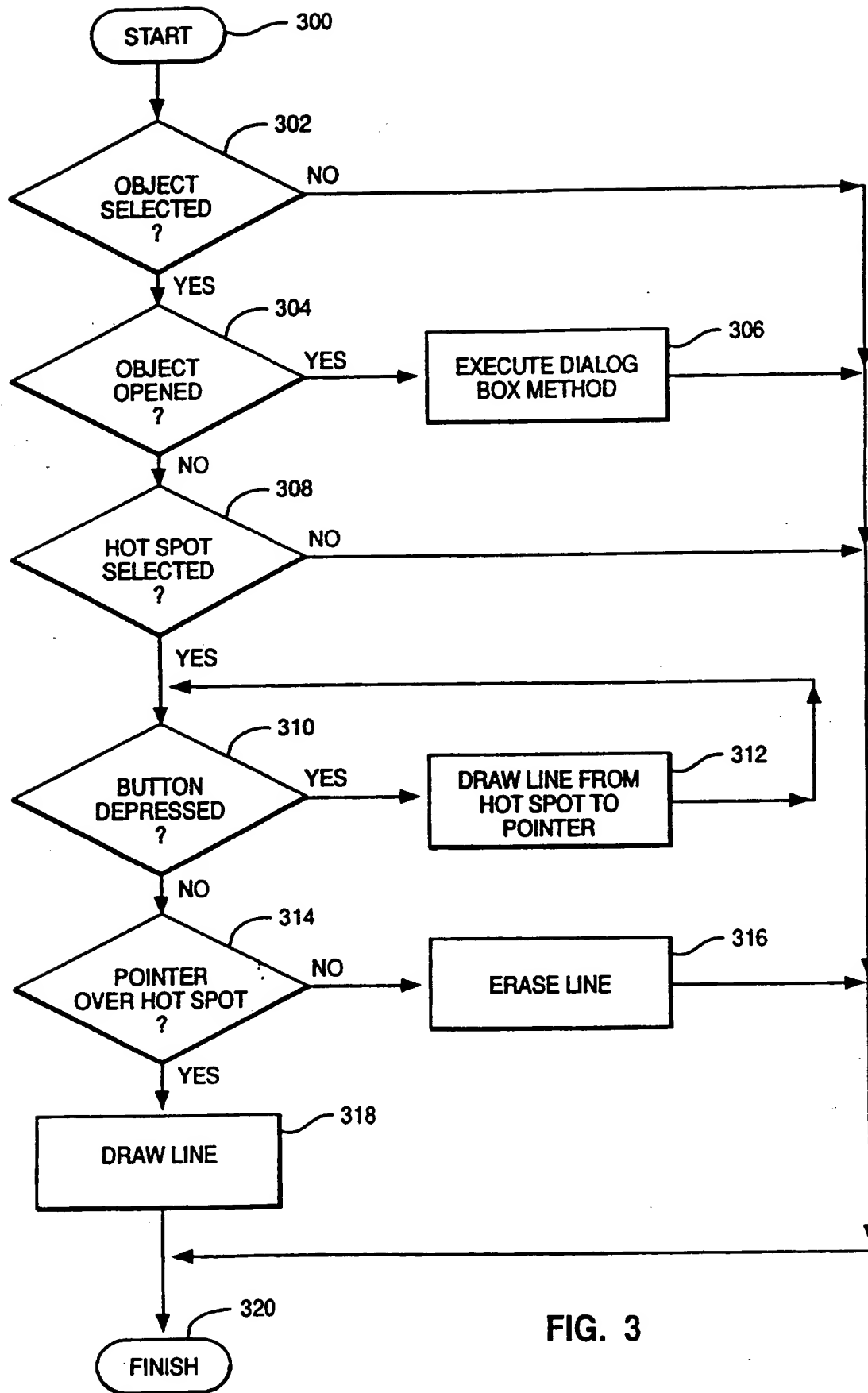


FIG. 3

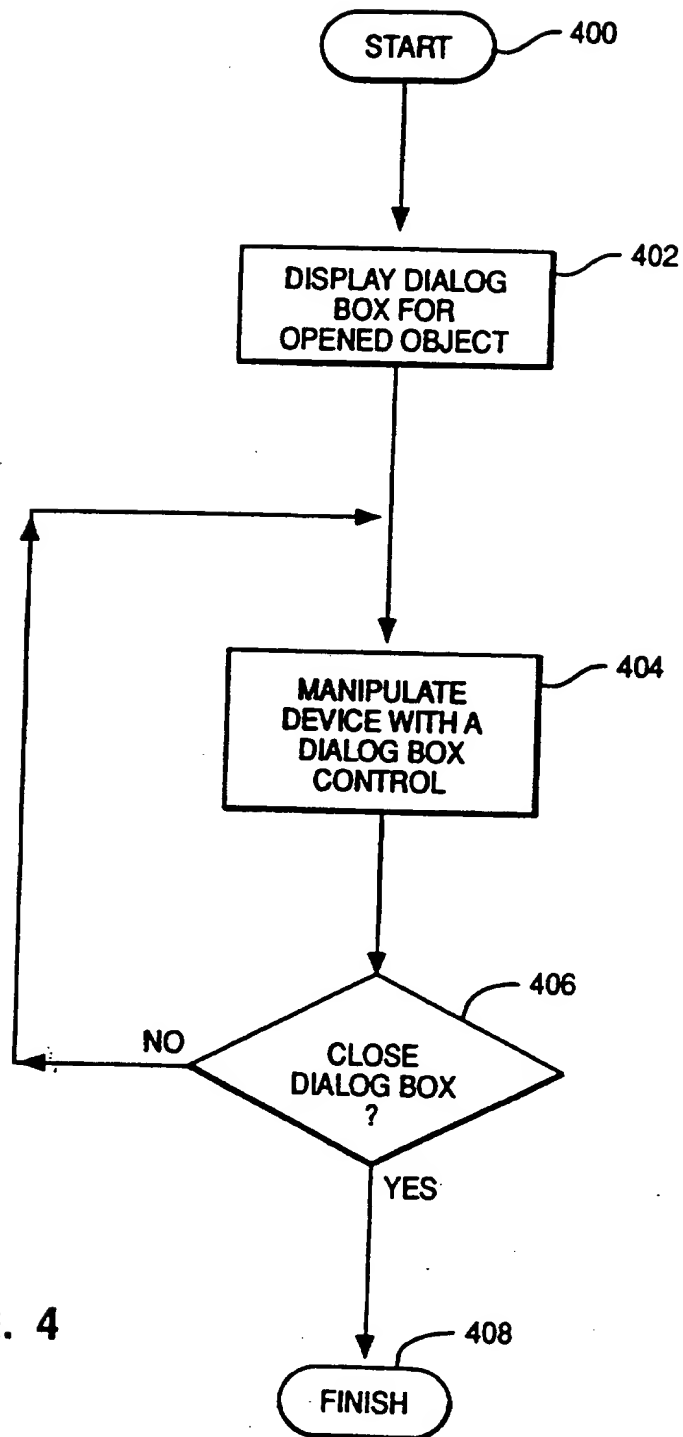
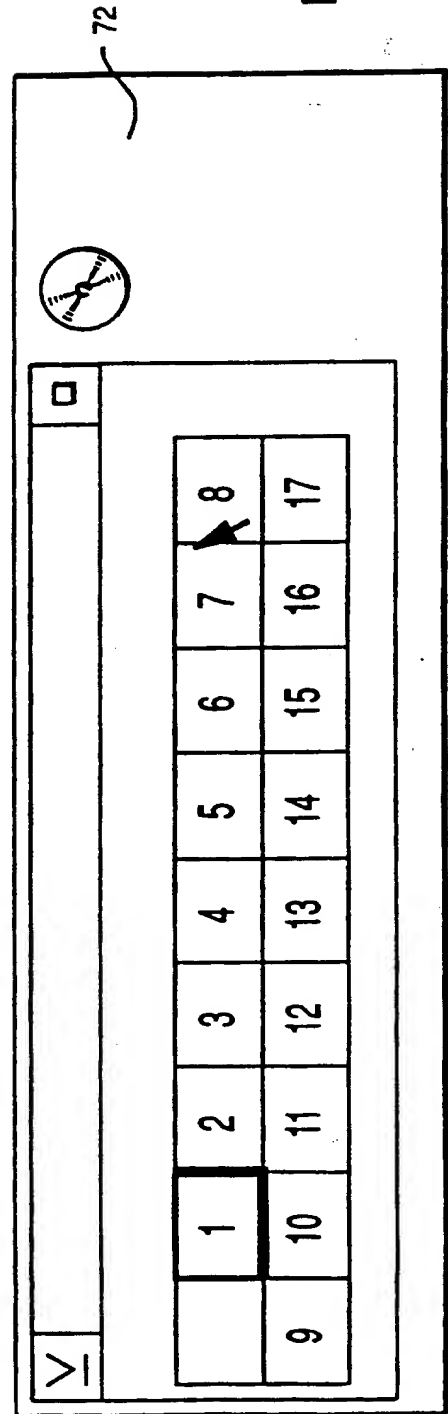
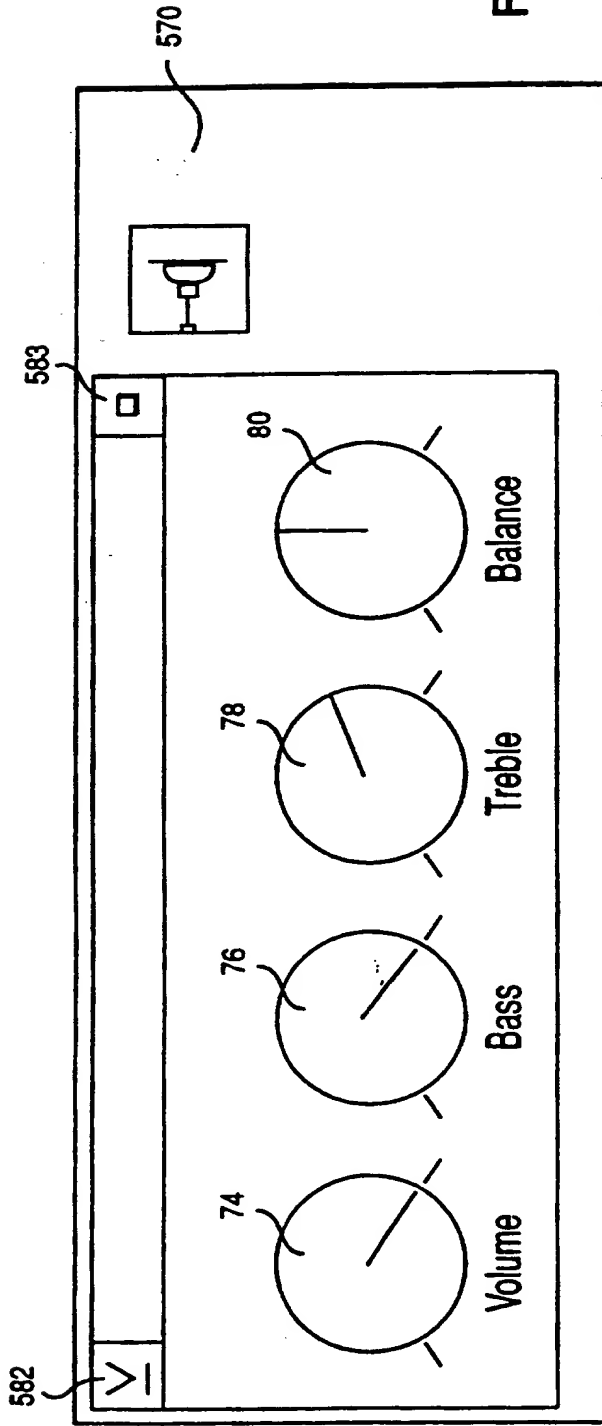


FIG. 4



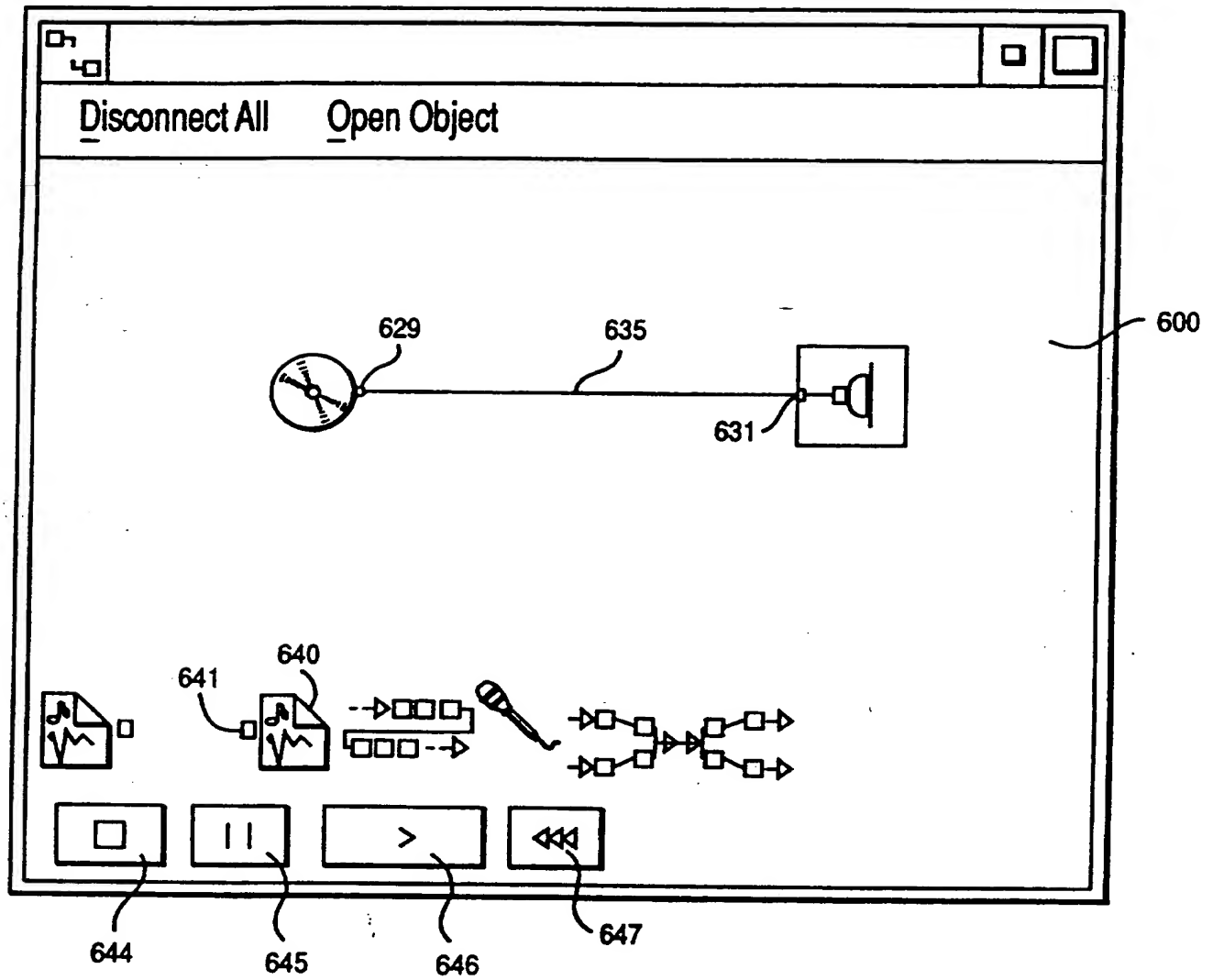


FIG. 6

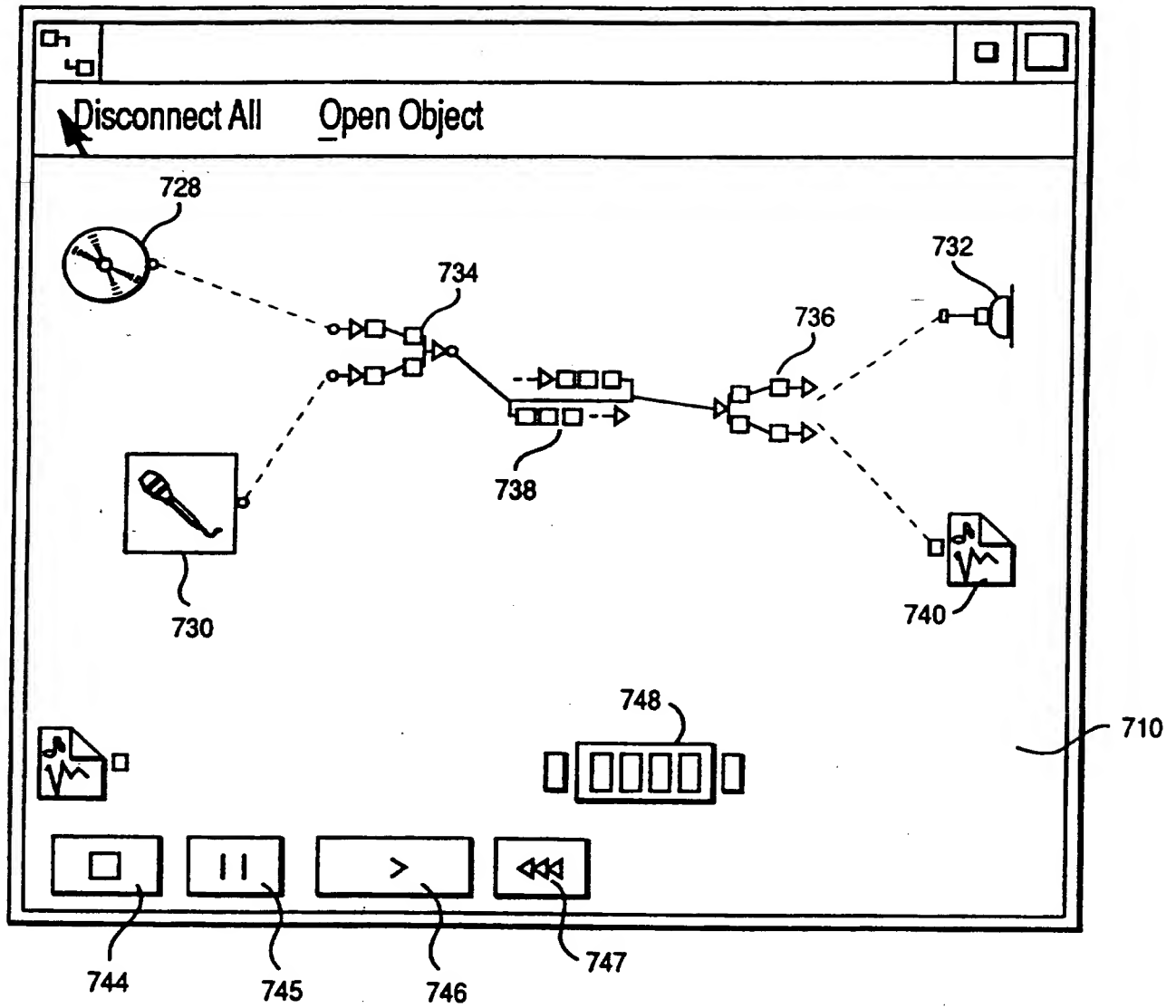


FIG. 7

AN OBJECT ORIENTED INTERFACE FOR CONTROLLING MULTIMEDIA DEVICES

FIELD OF THE INVENTION

5       The present invention relates to a multimedia computer system and more particularly to an object-oriented graphical user interface for controlling and connecting multimedia devices.

BACKGROUND OF THE INVENTION

10       In recent years there has been a move toward creating and using computer programs in an object-oriented environment. Such an environment utilizes the concept of objects to generally represent items. Each object comprises both data and operations, called methods, for manipulating the data. Such data and operations can be treated as a unit. One  
15       characteristic of object-oriented programming is that the data included in an object can only be accessed by means of the included methods, i.e. the data can only be changed in a predefined and consistent manner by programs which interact with the object. This characteristic is referred to as  
20       encapsulation.

25       The items that can be represented by objects, include common real world office objects, such as telephones and speakers, as well as computer peripherals, computer programs, and documents. An example of such an object (hereinafter referred to as an application object) is one which encapsulates the text of a word processing document. The object might contain methods which implement the ability of the object to automatically invoke a word processing program and to read itself into that word processing program upon selection of the object.

30       In an object-oriented graphical user interface, each application object typically includes a method for displaying a representation of the object on a visual display device, such as a monitor. Generally, this representation is in the form of an icon, which is a generally a bitmap of predetermined size where the appearance of the icon is defined by the  
35       bitmap.

40       Typically, in a graphical user interface, the selection of an object icon is performed using a pointing device, such as a mouse. The mouse allows a user to point to an icon by generating and controlling a pointer graphic on the display. When the pointing device is moved, the pointer

graphic also moves so that the user can move the pointer graphic over an icon on the display. Once the pointer graphic is positioned over the desired icon, the icon can be "selected" by operating a button or other selection device.

5

Alternatively, the icons may be moved on the display by use of the pointing device and selection button. In this manner of operating called "drag and drop", the icon is first selected as described above and then the pointing device is moved while the selection button is held operated. This combinations of actions causes the icon to follow the pointer graphic across the display device. When the selection button is released, the icon appears in a new position. In this manner one icon may be dragged and dropped on top of another icon to cause two underlying objects to interact.

10

15

The pointer and selection devices can be used in an object-oriented graphical interface to invoke the methods of the application objects which correspond to the displayed icons. For example, when a particular icon is selected, the interface typically sends a message to the underlying object informing it that its icon has been selected. In response to this message, a predetermined one of the corresponding object methods may be invoked. Similarly, double-clicking on an icon (positioning the pointer graphic over the icon and operating a selection button twice within a predetermined interval of time) may generate another message which is used to invoke another of the object methods.

20

25

Still other techniques may be used to invoke object methods. For example, selected areas of the icons may be designated as "hot spots". The hot spots of two icons can be connected together by a line segment on the display. This connection, in turn, causes messages to be forwarded to the underlying objects, which, in turn, cause the objects to execute selected methods.

30

35

40

Along with the movement towards object oriented environments, multimedia has been a rapidly emerging field in personal computing. Multimedia devices include, for example, compact disc players, laser disc players, VCRs, audio devices and digital audio tape players. In a multimedia system, these devices are typically connected either directly or indirectly to a computer system which includes a visual display as well as audio speakers driven by a specialized sound board. The computer digitizes and stores both the visual and audio information and controls the operation of the various devices to form an integrated system which can store and

simultaneously playback various information segments. Thus, it is possible to display digital video on the display screen while playing a synchronized digitized audio segment through the speakers.

5 Historically, each of the multimedia devices was operated as a stand-alone component. At first the user controlled each device from a physical control panel mounted on the device. Later, the devices became more integrated into the computer system and it was possible to control the devices by using commands entered into the computer. With the advent of  
10 graphical user interfaces, the computer was able to generate a control panel graphic on the screen which could be used to control the underlying device.

15 However, the devices still operated in an isolated fashion. For example, if a user desired to add a reverberation sound effect to a sound segment, it was necessary to first record the sound segment using a recorder, then digitize the segment and store it in a sound file. Then the sound file could be retrieved and edited to add the reverberation effect. Finally, the edited file could be played through a playback device.

20 Consequently, the power of multimedia systems has yet to be fully realized since the methods for controlling the devices in the system has essentially been limited to emulating device operation in a stand-alone, non-integrated prior art mode.

25 SUMMARY OF THE INVENTION

The present invention allows a user to control multimedia devices in a more human centric object oriented manner.

30 Another advantage of the present invention is the ability to provide a computer system capable of dynamically connecting and controlling multimedia devices.

35 Another advantage of the present invention is to allow a user to control separate multimedia devices in a user-friendly, integrated environment.

40 Still another advantage of the present invention is to provide an intuitive connection mechanism for connecting multimedia devices together to process a multimedia file in real time.

According to the present invention, a computer system connects and controls a plurality of multimedia devices, in response to a user's manipulations of icons indicative of the devices on an graphical user interface, the user can connect and control the multimedia devices as desired by simply manipulating icons representative of the multimedia devices displayed on the graphical user interface.

The present invention is applicable to both video, audio and other forms of data, such as haptic sensor data, and allows the user to easily connect multimedia devices, through a computer system, by simply selecting the icon indicative of a first multimedia device to be connected with a mouse pointing device, and dragging the mouse cursor to the icon indicative of a second multimedia device for linking. A linking line is drawn on the display as the mouse cursor moves, until the user clicks on the icon indicative of the second multimedia device he wishes to connect the first multimedia device to. As an example, the user may click on a compact disc player icon and link the compact disc player icon to a speakers icon on the graphical user interface. The computer will then perform the necessary internal hardware connections to implement the connection of the compact disc player and speakers as shown on the graphical user interface.

Once connected, a user can control the multimedia devices through the graphical user interface displayed on a computer display. Each separate device may be controlled by selecting its associated icon, in order to open a settings mechanism. The settings mechanism, which may be implemented with a dialog box, allows the user to directly control the device through the graphical user interface. As an example, the user can control the compact disc player and the speakers, or any other multimedia device in the system, by simply selecting the device's icon and manipulating the dialog box controls, each of which sets a device controllable parameter (e.g., volume, balance, bass, ....) displayed on the graphical user interface.

The present invention also allows a user to control the presentation of the multimedia information from the system in near real-time. As an example, if the system includes a graphics equalizer through which an audio signal is routed before being output to a set of speakers, a user may adjust the equalizer's frequency response via the graphical user interface to preprocess the audio before presentation. This preprocessing may be performed by simply selecting the icon associated with the equalizer and opening the associated object which may include a dialog box through which

the user can control the equalizer to preprocess the sound as it is being played back through the speakers.

5 In addition, the user can further edit (i.e., preprocess) the signals using a filter, a mixer or other well known devices. As used herein, the term "preprocess" includes any signal processing step which alters the characteristics of a received data signal. As an example, preprocessing may including mixing a received data signal with another signal, or filtering or delaying a received signal before providing the processed signal to a user.

10 Along with connecting and controlling various multimedia devices, the user can also disconnect a device from the system, by simply disconnecting the icon shown connected on the graphical user interface to the system. The computer system will then perform all the necessary hardware disconnections to actually implement the disconnection.

15 The present invention allows a user to quickly and easily interconnect a plurality of multimedia devices, via a graphical user interface, into an operable multimedia system and control the devices via a single, graphical user interface.

#### BRIEF DESCRIPTION OF THE DRAWINGS

20 The above and further advantages of the invention may be better understood by referring to the following description in conjunction with the accompanying drawings in which:

25 Figure 1 illustrates a functional embodiment of a data processing system for implementing the present invention;

30 Figure 2 illustrates an example of a screen display generated by an object-oriented graphical user interface displaying the various selectable icons, some of which represent multimedia devices;

35 Figure 3 is an illustrative flowchart of a routine for opening and selecting an icon displayed on the object-oriented graphical user interface;

40 Figure 4 is an illustrative flowchart of a subroutine for manipulating the multimedia device associated with a selected icon;

Figures 5A and 5B illustrate dialog box screen displays on the graphical user interface for a speaker and a compact disc player respectively;

Figure 6 illustrates two icons connected and displayed on the object oriented graphical user interface; and

Figure 7 illustrates a plurality of icons connected together and displayed on the graphical user interface to form a karaoke machine.

#### DETAILED DESCRIPTION

Figure 1 illustrates the system architecture for a conventional computer system, such as an IBM PS/2® personal computer (PC). The exemplary computer system of Figure 1 is for descriptive purposes only. Though the description below may refer to terms commonly used in describing particular computer systems, such as an IBM PS/2 PC, the description and concepts equally apply to other systems, including systems having architectures dissimilar to Fig. 1.

The exemplary computer 100 includes a central processing unit (CPU) 105, which may include a conventional microprocessor; a system random access memory (RAM) 110 for temporary storage of information and a read only memory (ROM) 1115 for permanent storage of information. A memory controller 120 is provided for controlling system RAM 110; a bus controller 125 is provided for controlling bus 130; and an interrupt controller 135 is used for receiving and processing various interrupt signals.

Mass storage may be provided by a diskette 142, a CD-ROM disk 147 or a hard disk 152. The diskette 142 can be inserted into a diskette drive 141, which is, in turn, connected to bus 130 by a controller 110. Similarly, the CD-ROM disk 147 can be inserted into a CD-ROM drive 146, which is also connected by a controller 145 to bus 130. Finally, hard disks 152 are part of a fixed disk drive 151, which is connected to bus 130 by controller 150.

Input and output to computer system 100 is provided by a number of devices. For example, a keyboard and mouse controller 155 connects to bus 130 for controlling a keyboard input device 156 and a mouse input device 157. A DMA controller 160 is provided for performing direct memory access to system RAM 110. A visual display is generated by a video controller 165,

which controls a video output display 170. Display 170, under the control of the computer system 100, generates a two dimensional array of picture elements (pixels), which may be independently controlled to form an image. Other input and output devices, such as an audio, video or special purpose cards may be connected to the system through expansion slot 190, and are shown here as multimedia devices 80-83.

The computer 100 is generally controlled and coordinated by operating system software, such as the OS/2<sup>®</sup> operating system, available from the International Business Machines Corporation (IBM), Boca Raton, Florida. Operating systems provide resource management throughout a computer system, including such tasks as process execution and scheduling memory management, file system services, networking and scheduling and I/O services, and user interface presentation. User applications, such as editors and spread sheets, directly or indirectly, rely on these and other capabilities of the operating system.

In addition, modern computers are increasingly using multimedia techniques, which store, organize, and present various forms of data, including textual data, digital audio data, digital video data, and digital music data (e.g., MIDI). For example, a computer using multimedia techniques may play back video data and audio data to produce a movie clip video sequence on display 170 with synchronized audio output from multimedia devices 80-83.

In order to the control the multimedia devices 80-83 in accordance with the present invention, a user controls the computer 100 via an object-oriented graphical user interface generated by the operating system, controlled by the application program and presented on the display monitor 170.

As will be understood by those skilled in the art, Object-Oriented Programming (OOP) techniques involve the definition, creation, use and destruction of "objects". These objects are software entities comprising data elements and routines, or methods, which manipulate the data elements. The data and related functions are treated by the software as an entity and can be created, used and deleted as if they were a single item. Together, the data and functions enable objects to model virtually any real-world entity in terms of its characteristics, which can be represented by the data elements, and its behaviour, which can be represented by its data manipulation functions.

Objects are defined by creating "classes" which are not objects themselves, but which act as templates that instruct a program compiler how to construct the actual object. A class may, for example, specify the number and type of data variables and the steps involved in the functions which manipulate the data. An object is actually created in an object-oriented program at run-time by means of a special function called a constructor which uses the corresponding class definition and additional information, such as arguments provided during object creation, to construct the object. Memory space is allocated for the object data and methods at the time of creation. Likewise objects are destroyed at run-time by a special function called a destructor. When an object is destroyed, the allocated memory is released. Objects may be used by using their data and invoking their functions.

The principle benefits of object-oriented programming techniques arise out of three basic principles; encapsulation, polymorphism and inheritance. More specifically, objects can be designed to hide, or encapsulate, all, or a portion of, the internal data structure and the internal functions. More particularly, during program design, a program developer can define objects in which all or some of the data variables and all or some of the related functions are considered "private" or for use only by the object itself. Other data or functions can be declared "public" or available for use by other programs. Access to the private variables by other programs can be controlled by defining public functions for an object which access the object's private data. The public functions form a controlled and consistent interface between the private data and the "outside" world. Any attempt to write program code which directly accesses the private variables causes the compiler to generate an error during program compilation which error stops the compilation process and prevents the program from being run.

Polymorphism is a concept which allows objects and functions which have the same overall format, but which work with different data, to function differently in order to produce consistent results. For example, an addition function may be defined as variable A plus variable B ( $A+B$ ) and this same format can be used whether the A and B are numbers, characters or dollars and cents. However, the actual program code which performs the addition may differ widely depending on the type of variables that comprise A and B. Polymorphism allows three separate function definitions to be written, one for each type of variable (numbers, characters and dollars). After the functions have been defined, a program can later refer to the addition function by its common format ( $A+B$ ) and, during compilation, the

C++ compiler will determine which of the three functions is actually being used by examining the variable types. The compiler will then substitute the proper function code. Polymorphism allows similar functions which produce analogous results to be "grouped" in the program source code to produce a more logical and clear program flow.

The third principle which underlies object-oriented programming is inheritance, which allows program developers to easily reuse pre-existing programs and to avoid creating software from scratch. The principle of inheritance allows a software developer to declare classes (and the objects which are later created from them) as related. Specifically, classes may be designated as subclasses of other base classes. A subclass "inherits" and has access to all of the public functions of its base classes just as if these function appeared in the subclass. Alternatively, a subclass can override some or all of its inherited functions or may modify some or all of its inherited functions merely by defining a new function with the same form (overriding or modification does not alter the function in the base class, but merely modifies the use of the function in the subclass). The creation of a new subclass which has some of the functionality (with selective modification) of another class allows software developers to easily customize existing code to meet their particular needs.

Although object-oriented programming offers significant improvements over other programming concepts, program development still requires significant outlays of time and effort, especially if no pre-existing software programs are available for modification. Consequently, a prior art approach has been to provide a program developer with a set of pre-defined, interconnected classes which create a set of objects and additional miscellaneous routines that are all directed to performing commonly-encountered tasks in a particular environment. Such pre-defined classes and libraries are typically called "application frameworks" and essentially provide a pre-fabricated structure for a working application.

For example, an application framework for a user interface might provide a set of pre-defined graphic interface objects which create windows, scroll bars, menus, etc. and provide the support and "default" behaviour for these graphic interface objects. Since application frameworks are based on object-oriented techniques, the pre-defined classes can be used as base classes and the built-in default behaviour can be inherited by developer-defined subclasses and either modified or overridden to allow developers to extend the framework and create customized solutions in a particular area

of expertise. This object-oriented approach provides a major advantage over traditional programming since the programmer is not changing the original program, but rather extending the capabilities of the original program. In addition, developers are not blindly working through layers of code because the framework provides architectural guidance and modelling and; at the same time, frees the developers to supply specific actions unique to the problem domain.

There are many kinds of application frameworks available, depending on the level of the system involved and the kind of problem to be solved. The types of frameworks range from high-level application frameworks that assist in developing a user interface, to lower-level frameworks that provide basic system software services such as communications, printing, file systems support, graphics, etc. Commercial examples of application frameworks include MacApp by Apple Computer, Inc., Bedrock by Symantec Corporation and OWL by Borland International, Inc.

In accordance with the invention, a multimedia application framework is provided which includes a plurality of multimedia objects. A particular multimedia object is associated with each type of multimedia device. Each multimedia object includes an icon which is displayed when the object is created. Each multimedia icon also includes internal methods which control the associated multimedia device, both to set the device operating parameters, direct input to and output from the device and control operation of the device. The relationship of the objects to their associated devices is discussed in detail below.

Figure 2 illustrates an exemplary display 226 generated by an object-oriented graphical user interface. In addition to the usual window area 210 associated with the application program, representations of various selectable objects are displayed in the form of icons 228-238. Each of icons 228-238 is associated with an underlying object which, in turn, controls a multimedia device. Such devices include, for example, a compact disc player, a microphone, speakers, a mixer, a splitter, and a reverberation filter, corresponding icons 228, 230, 232, 234, 236 and 238, respectively. Other multimedia devices such as a digital audio tape player/recorders may also have an associated object with a different icon (not shown). Each of the icons 228-238 has a connection point or "hot spot" which can be used, as described in detail below to connect the icons which, in turn, effects a connection between the underlying multimedia devices. For example, compact disk icon 228 has a hot spot

229 and speaker icon 232 has a hot spot 231. Some icons have only one hot spot indicating that they only receive or generate data, while other icons have two hot spots indicating that they both receive an input and generate an output. For example, reverberation icon 238 has an input hot spot 237 and an output hot spot 239.

In a conventional manner, the graphical user interface also includes pull down menus 242, 243 and common control icons 244-247 for the various multimedia devices represented by the icons. The control icons include stop 244, pause 245, play 246 and rewind 247 icons which the user can select via the mouse 114 or the keyboard 116. Additional icons may also be used to represent multimedia files, or a recording device to record data in the file, such as output file icon 240.

The underlying multimedia devices are controlled by manipulating the displayed icons in order to control the associated object. The manipulation of an object can take the form of selecting the associated object, opening the object and connecting two objects to cause the output of one object to be provided to the input of another object. In addition, a multimedia device may also be controlled by the common control icons 244-247. These latter icons operate in conjunction with a selected object to control the underlying device in a manner that will hereinafter be discussed.

Figure 3 is a flowchart of a routine for opening and selecting an object displayed on the graphical user interface. The routine shown in Figure 3 and those subsequently disclosed herein, are preferably part of each object. In accordance with normal object-oriented programming techniques, the routine shown in Figure 3 would be included in a parent or base class which includes routines common to all multimedia objects.

The routine starts at step 300. In step 302 a determination is made whether the user has selected an object. As previously mentioned, an object may be selected in a variety of ways. One typical way would be to place the pointer graphic on the icon associated with the object and press the selection button. The latter two actions cause the graphical user interface to forward a message to the object associated with the icon. In response to the message, the object may change the appearance of the icon to visually indicate the selection.

If the object has not been selected, the routine ends in step 320. Alternatively, if the object has been selected as determined in step 302, the routine proceeds to step 304 in which a determination is made whether the object has been opened. Opening an object can be as simple as  
5 placing the pointer graphic over the icon and selecting the object, for example, double-clicking the selection button on a mouse. Other actions may also be used to open the associated object. For example, the object can be selected and then opened using the Open Object menu command 243.

10 Once a user has commanded the object to open as determined in step 304, a subroutine 306 is called to display a dialog box associated with the object the user has opened. As will hereinafter be explained in detail, such a dialog box may include a plurality of user manipulatable graphic controls such as push-buttons, dials, sliders, potentiometers,  
15 etc. Each of these controls calls a method in the object which adjusts a parameter of the underlying multimedia device.

20 Figure 4 is a more detailed flowchart of the subroutine 306 for manipulating the multimedia device parameters associated with the object the user has opened. In general, each multimedia object 228-238 will have its own dialog box (or boxes) which is specific to the underlying multimedia object type and displayed on the graphical user interface when the user opens the object. The dialog box routine starts in step 400 and proceeds to step 402. In step 402 a dialog box routine in the opened  
25 object is called to display the dialog box screen for the opened object. In general, the dialog box routine creates a dialog box object which then generates the dialog box screen display.

30 In step 404, a user sets a parameter of the multimedia device associated with the opened object as commanded by the user's manipulation of one of the control icons within the dialog box. The operation and control of such icons is well-known to those skilled in the art. Each control icon is, in turn, associated with a control object which contains methods for setting parameters of the associated object. Accordingly,  
35 manipulating one of the controls causes messages to be sent to the control object which then uses one of the included methods to actually set the device parameter.

40 After a control is manipulated, a determination is made in step 406, whether the dialog box should be closed. Generally the box is closed in response to a command generated by the user. Such a command

may, for example, take the form of operation of an "OK" push-button, a "Cancel" push-button or selection of the dialog box window system box. When the dialog box is closed the dialog box object and all control objects are destroyed.

If no command has been received to close the dialog box, then from step 406, the routine loops and re-executes steps 404 and 406 until the user closes the dialog box.

Figures 5A and 5B show illustrative dialog boxes 570, 572 which might be used to set parameters of a amplifier/speaker device and a compact disc player device, respectively. The dialog box 570 would be displayed when the speaker object (represented by the speaker icon 232 in Figure 2) was opened and the dialog box 572 would be displayed when the compact disc player object (represented by the compact disk icon 228 in Figure 2) was opened.

The control objects in the speaker dialog box 570 are represented by graphic potentiometer icons for volume 74, bass 76, treble 78 and balance 80. The user can manipulate these controls in order to control (i.e., preprocess) the sound from the associated amplifier/speaker by typically selecting the control and dragging the control to a desired position. As mentioned the manipulation of a control icon, for example, the volume icon 74, sends messages to the associated control object. In response, the control object uses one of its methods to control the multimedia device (Figure 1) to provide the necessary commands to the speaker amplifier in order to adjust the speaker volume.

Similarly, Figure 5B shows a dialog box 572 which might be used to control an illustrative compact disk player represented by compact disk icon 228 (Figure 2). The control icons associated with dialog box 572 are a plurality of push-buttons 590-598. When one of push-buttons 590-598 is selected by positioning the pointer graphic 592 over the selected button, a method is called in the associated control object which causes one track on the compact disk to be selected for playback.

Returning to Figure 3, if a determination is made in decision block 304 that the user has not opened an object, a further determination is made in decision block 308 whether a hot spot on the object icon has been selected which indicates whether the user has selected an object for connection to another object.

If a hot spot has been selected, the routine proceeds to step 310 where another determination is made whether the selection button remains depressed. If the button remains depressed, a drag operation is proceeding. In this latter case, a method is called in the underlying object which causes a line to be drawn from the selected hot spot to the position of the pointer graphic as indicated in step 312. This line can also be drawn by creating a separate connection object and calling a method in the connection object to draw the line. The line connecting the selected hot spot to the pointer graphic continues to be drawn as long as the selection button remains depressed.

If, in step 310, a determination is made that the selection button has been released, decision block 314 makes a further decision whether the pointer graphic is positioned over a hot spot on another target icon. If so, the routine proceeds to step 318 where a line is drawn between the hot spot in the source icon to the hot spot in the target icon indicating that the two underlying devices are to be connected together. In addition, a method in the target object is called which returns the address of the target object data input. This address is then provided to the source object which causes the output of the underlying multimedia device to be directed to the input of the target device.

Alternatively, if a determination is made in decision block 314 that the pointer graphic is not located over a hot spot on the target icon when the selection button is released, the line connecting the selected hot spot on the source icon and the pointer graphic is erased in step 316 and the routine finishes in step 320.

To illustrate how a user can quickly and easily connect various multimedia devices using an object oriented graphical user interface, Figure 6 illustrates an object oriented graphical user interface 600 displaying a compact disk player icon 628 operatively connected to a speaker icon 632. To perform this connection, the user simply positions the pointer graphic on the hot spot 629 of compact disc player icon 628 and depresses the selection button to select the hot spot. Then, keeping the selection button depressed, the user moves the pointing tool and drags the pointer graphic to the hot spot 631 of speaker icon 632.

When the selection button is released, a line 635 is drawn connecting the two hot spots. The connection routine illustrated in Figure 3 then calls the necessary methods in the underlying objects to

actually connect the output of the compact disc player multimedia device to the input of the speaker amplifier to allow the compact disk player to play through the speaker. Alternatively, the output of compact disk player 628 could be directed to a sound file 640 by connecting the output hot spot 629 of the compact disk icon 628 to the input hot spot 641 of the sound file. This latter connection would cause a new sound file to be opened and a recording device to be started to receive input for the compact disk player device.

Once the connection between the compact disk player and the speaker is established, the player can be controlled by means of the control icons 644-647. For example, the compact disk player icon can be selected by positioning the pointer graphic over it and depressing the selection button. Subsequently, the play icon 646 can be selected, in turn, causing a method in the underlying control object to be executed and start the compact disk player device. In a similar manner, the stop icon 644 can be selected to stop the player. The player can also be paused by selecting the pause icon 645. The rewind icon 647 can be selected to return the player to the beginning of the selection being played back.

The present invention is not limited to operatively connecting only two multimedia devices, but can also connect any number of compatible multimedia devices to create a complete multimedia system. As an example, Figure 7 illustrates a plurality of icons on a graphical user interface 710 operatively connected to form a "karaoke" system. The object-oriented graphical user interface 710 illustrates the icons 728-740 of the various karaoke system components operatively connected. Icons 728 and 730, indicative of a compact disc player device and a microphone device respectively are connected to a mixer icon 734 which causes the background music output from the compact disc player device to be mixed or combined with the user/singer's voice output from the microphone device. The mixed signal is then input to a reverberation filter represented by icon 738 and routed to a splitter device represented by icon 736 which provides musical output to the speaker represented by icon 732 and a recording device/output file 740 so that the singer can have a souvenir of his recording. Other devices, such as the equalizer 748 can be added by connecting them in series with the remaining components.

Components can be disconnected by simply selecting the connecting line by positioning the pointer graphic over the line and pressing the

selection button. Once selected, the line can be deleted by operating a further control such as the DELETE key. Operation of the DELETE key then calls one of the object methods in the underlying multimedia objects to break the connection between the multimedia devices.

5 If the user desires to increase the volume output from the karaoke system speaker, he would simply open the speaker object represented by icon 732 and the speaker dialog box 572 (Figure 5A) would appear on the graphical user interface. The user may then increase the volume or  
10 adjust the audio as desired. Similarly, to play another song for the next karaoke singer, the user would simply open the compact disc player object represented by icon 728 and the compact disc player dialog box 570 (Figure 5B) would appear to allow the user to select, via the graphical user interface, the track on the compact disk to be played next. Once a  
15 song is queued up and the singer is ready to go, the user can start the song by selecting the play icon 746. The other control icons, 744, 745 and 747 may also be selected to stop, pause and rewind the player, respectively.

20 The term "preprocess" as used herein includes any technique which changes the characteristics of an incoming signal or signals, for example, mixing two or more signals, adjusting the audio or video characteristics of a signal, or, in general, any other technique for changing the characteristics of an audio, video, or other signal in  
25 either analog or digital format.

Along with connecting the various devices in the system, the present invention can perform the necessary synchronization between the devices. In addition, once a system, such as the karaoke system is  
30 defined, it is contemplated that the present invention permits the user to define and save the entire system, including various multimedia device connections and settings, as an object, complete with an icon representation of the object. In this manner, the connection of the devices and settings to form the karaoke machine can be performed again  
35 by simply selecting a user-defined karaoke machine icon 748.

Although the present invention has been shown and described with respect to a preferred embodiment thereof, it should be understood by those skilled in the art, that various other changes, omissions and  
40 additions to the form and detail thereof, may be made therein without departing from the spirit and scope of the present invention. For

example, the present invention is not limited to use with a mouse. A track ball, light pen, touch screen or any like device can be used by a user to manipulate the icons displayed on the graphical user interface.

5           In addition, although the illustrative embodiment of the multimedia computer system is a personal computer, the invention is clearly not so limited. It is contemplated, that computer systems other than personal computers may also use the method and apparatus of the present invention to operatively connect and control a plurality of  
10 multimedia devices via an object oriented graphical user interface.

          In an alternate embodiment, the invention may be implemented as a computer program product for use with a computer system. Such  
15 implementation may comprise a series of computer readable instructions either fixed on a tangible medium, such as a computer readable media, e.g. diskette 142, CD-ROM 147, ROM 115, or fixed disk 152 (Fig. 1), or transmittable to a computer system, via a modem or other interface device, over either a tangible medium, including but not limited to optical or analog communications lines, or intangibly using wireless  
20 techniques, including but not limited to microwave, infrared or other transmission techniques. The series of computer readable instructions embodies all or part of the functionality previously described herein with respect to the invention. Those skilled in the art will appreciate that such computer readable instructions can be written in a number of programming languages for use with many computer architectures or  
25 operating systems. Further, such instructions may be stored using any memory technology, present or future, including, but not limited to, semiconductor, magnetic, optical or other memory devices, or transmitted using any communications technology, present or future, including but not  
30 limited to optical, infrared, microwave, or other transmission technologies. It is contemplated that such a computer program product may be distributed as a removable media with accompanying printed or electronic documentation, e.g., shrink wrapped software; preloaded with a computer system, e.g., on system ROM or fixed disk, or distributed from a server or electronic bulletin board over a network, e.g., the Internet or  
35 World Wide Web.

CLAIMS

1. In a computer system having a display device and a graphical user interface for generating icon displays on the display device, an object-oriented method for controlling a plurality of multimedia devices to process multimedia data, the method comprising the steps of:

A. creating a multimedia object associated with each of the plurality of multimedia devices, the object having a plurality of object methods for controlling the associated multimedia device and a method for controlling the graphical user interface to generate an icon representative of the object on the display;

B. controlling the graphical user interface to visually connect a first icon representing a first multimedia object to a second icon representing a second multimedia object; and

C. calling at least one of the plurality of object methods in the first and second multimedia objects to direct the multimedia data from a first multimedia device associated with the first multimedia object to a second multimedia device associated with the second multimedia object.

2. The method of claim 1 wherein at least one of the plurality of multimedia devices has a device parameter which controls the processing of the multimedia data by the at least one multimedia device and wherein the method further comprises the steps of:

D. creating a control object having a first control object method for setting the device parameter and a second control object method for displaying a control icon; and

E. calling the control object method to set the device parameter.

3. The method of claim 2 wherein step D comprises the steps of:

D1. opening a multimedia object associated with the at least one multimedia device;

D2. creating a dialog box object on the graphical user interface in response to the opening of the multimedia object in step D1;

5 D3. creating the control object and calling the first control object method to display the control icon; and

10 D4. calling the second control object method to set the device parameter in response to user manipulation of the control icon.

4. The method of claim 1 wherein step B comprises the step of:

15 B1. calling one of the plurality of object methods in the first and the second multimedia objects to control the graphical user interface to draw a line connecting the first icon to the second icon.

20 5. The method of claim 1 wherein step B further comprises the steps of:

25 B2. creating a connection object having an object method for controlling the graphical user interface to draw a line connecting the first icon to the second icon; and

B3. calling the connection object method.

30 6. A computer program product for use with a computer system having a display device for displaying a graphical user interface, the computer program product comprising:

35 a computer usable medium having computer readable program code means embodied thereon for controlling a plurality of multimedia devices, the computer usable media comprising:

40 program code means for creating a object associated for each of the plurality of multimedia devices, the object having a plurality of object methods for controlling the associated multimedia device and a method for controlling the graphical user interface to generate an icon representative of the object on the display;

program code means for controlling the graphical user interface to visually connect a first icon representing a first multimedia object to a second icon representing a second multimedia object; and

5           program code means for calling at least one of the plurality of object methods in the first and second multimedia objects to direct the multimedia data from a first multimedia device associated with the first multimedia object to a second multimedia device associated with the second multimedia object.

10           7.     The computer program product of claim 6 wherein at least one of the plurality of multimedia devices has a device parameter which controls the processing of the multimedia data by the at least one multimedia device and wherein the program code means for creating an object further  
15     comprises:

          program code means for creating a control object having a first control object method for setting the device parameter and a second control object method for displaying a control icon; and

20           program code means for calling the control object method to set the device parameter.

25           8.     The computer program product of claim 7 wherein said program code means for creating further comprises:

          program code means for opening a multimedia object associated with the at least one multimedia device;

30           program code means for creating a dialog box object on the graphical user interface in response to the opening of the multimedia object;

35           program code means for creating the control object and calling the first control object method to display the control icon; and

          programs code means for calling the second control object method to set the device parameter in response to user manipulation of the control icon.

9. The computer program product of claim 6 wherein said program code means for controlling the user interface comprises:

5 program code means for calling one of the plurality of object methods in the first and the second multimedia objects to control the graphical user interface to draw a line connecting the first icon to the second icon.

10 10. In a computer system having a display device, a graphical user interface for generating icon displays on the display device, and a pointing device under control of an application program executing on the computer system, an apparatus for controlling a plurality of multimedia devices to process multimedia data comprising:

15 means controlled by the application program for creating a multimedia object associated with each of the plurality of multimedia devices, the object having at least one object method for controlling the associated multimedia device and a method for controlling the graphical user interface to generate an icon representative of the object  
20 on the display;

25 connection means, cooperating with the pointing device, for controlling the graphical user interface to visually connect first and second icons;

30 means responsive to the connection of said first icon, representing first multimedia object, to said second icon, representing a second multimedia object, for calling at least one object method for each of the first and second multimedia objects to direct the multimedia data from a first multimedia device associated with the first multimedia object to a second multimedia device associated with the second multimedia object.

35 11. The apparatus of claim 10 wherein at least one of the first and second multimedia devices has a device parameter which controls the processing of the multimedia data by the at least one multimedia device and wherein the apparatus further comprises:

40 means controlled by the pointing device for opening one of a plurality of multimedia objects;

means responsive to the opening of the one multimedia object for creating a control object having a first control object method for setting the device parameter and a second control object method for displaying a control icon; and

5

means controlled by the pointing device for calling the control object method to set the device parameter.

12. The apparatus of claim 11 wherein the creating means comprises:

10

means controlled by the pointing means for opening a multimedia object associated with the at least one multimedia device;

15

means responsive to the opening of the multimedia object for creating a dialog box object on the graphical user interface;

20

means responsive to the creation of the dialog box object for creating the control object and calling the first control object method to display the control icon; and

means controlled by the pointing device for calling the second control object method to set the device parameter in response to user manipulation of the control.



Application No: GB 9616653.3  
Claims searched: 1-5,10-12

Examiner: B G Western  
Date of search: 4 October 1996

**Patents Act 1977**  
**Search Report under Section 17**

**Databases searched:**

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.O): G4A AKS APX

Int Cl (Ed.6): G06F 3/023 3/033 9/44 17/30

Other: On-line databases: COMPUTER, INSPEC, WPI

**Documents considered to be relevant:**

Category	Identity of document and relevant passage			Relevant to claims
X	EP-0567699-A1	HEWLETT-PACKARD	N.b. pages 1-14	1,10
X	WO-95/08148-A1	TALIGENT	N.b. pages 1-13	1-5,10-12
X	WO-93/15455-A1	INTERACTIVE MEDIA	See whole document	1,2,4,5,10
X	US-5388264-A	TOBIAS et al	See whole document	1-5,10-12

X Document indicating lack of novelty or inventive step  
Y Document indicating lack of inventive step if combined with one or more other documents of same category.

& Member of the same patent family

A Document indicating technological background and/or state of the art.  
P Document published on or after the declared priority date but before the filing date of this invention.  
E Patent document published on or after, but with priority date earlier than, the filing date of this application.

This Page Blank (uspto)